iocage(8)

Michał Borysiak
Miłosz Kaniewski
Jarosław Żurek

Materiały do warsztatów

Obraz: https://bit.ly/2FZ7iRm

Skrypt: http://bit.do/eGHdm

WPROWADZENIE

Czym jest jail?

- jedna z możliwości wirtualizacji na poziomie systemu operacyjnego;
- pozwala zestawić własny sandboxowany userland, używając wspólnego kernela;
- “zaawansowany chroot(8)”;
- środowisko praktycznie nieodróżnialne od maszyny matki;
- wirtualizacja dostępu do systemu plików, użytkowników;
- każdy jail jest odseparowany i niezależny od innych z własną konfiguracją;

Czego nie można w jailu?

- nie można wykonywać połączeń poniżej TCP/UDP;
- nie można modyfikować kernela (sysctl);
- nie da się zarządzać interfejsem sieciowym z poziomu jaila;
- nie da się podejrzeć procesów hosta i innych jaili;
- nie można montować systemów plików;
- nie można dodawać urządzeń do `/dev`;
- nie można opuścić jaila;

Pierwszy jail skonfigurowany ręcznie (1 / 2)

```
mkdir ~/JAIL; cp /tmp/base.txz ~/JAIL; cd ~/JAIL      # ręcznie trzeba pobrać base :(
tar -xf base.txz                                      # wypakowanie base
touch /etc/jail.conf                                  # plik konfiguracyjny jaili *
jail -f /etc/jail.conf -c <name>                      # utworzenie jaila
```

 # Konfiguracja SSH

```
jexec <name> sysrc sshd_enable=”YES”                  # właczenie SSH przy starcie
jexec <name> vi /etc/ssh/sshd_config                  # edycja np. PermitRootLogin yes
jexec <name> service sshd start                       # start SSH

jail -f /etc/jail.conf -rc <name>                     # recreate jaila
```

przykład na następnym slajdzie

Pierwszy jail skonfigurowany ręcznie (2 / 2)

```
exec.start = "/bin/sh /etc/rc";
exec.stop = "/bin/sh /etc/rc.shutdown";
host.hostname = $name;
name {
    ip4.addr = "em0|10.0.70.145/16";
    path = /path/to/directory/with/JAIL;
    mount.devfs;
}
```

Czym jest iocage(8)?

Menedżer do jaili napisany w Pythonie, używający ZFS'a oraz VNET*, znacznie upraszczający tworzenie, konfigurację oraz zarządzanie jailami.

Przykładowe opcje do zarządzania :

- **create** / **destroy / start** / **stop** / **list**
- **fetch** - pobiera wskazany release
- **exec** - uruchamia polecenie w jailu
- **console** - wejście do jaila
- **clone** - pozwala sklonować dataset z jailem
- **update** / **upgrade** - umożliwia update FreeBSD w jailu do najnowszej / wybranej wersji
- **snaplist** / **snapshot** / **snapremove** - zarządzanie snapshotami datasetów per jail
- **get** / **set** - pobranie / ustawienie właściwości jaila np. sieci, hostname'u, quoty, mountpointów

*VNET - w pełni zwirtualizowany stos sieciowy, który jest izolowany per jail. domyślnie wyłączony, ale możliwy do konfiguracji np. **iocage set ip4_addr="vnet0|192.168.0.10/24" jail**

Rodzaje jaili

- **clone** `# iocage create -r 11.1-RELEASE`
  * klasyczny jail utworzony z pobranego release'u FreeBSD;
- **basejail** `# iocage create -r 11.1-RELEASE -b`
  * używają `mount_nullfs(8)` do udostępniania katalogów bazowych;
  * aktualizacja jednego userlandu uaktualnia pozostałe basejaile;
- **template**
  * umożliwia utworzenie bazowego jaila i na jego podstawie tworzone są kolejne;
- **empty** `# iocage create -e`
  * tworzony jest pusty jail, który pozwala na niestandardową konfigurację np. postawienie jaila z Linuxem;
- **thickjail** `# iocage create -T -r 11.1-RELEASE`
  * w pełni niezależne, dokładna kopia z obrazu (zajmują dużo miejsca);
  * idealne do synchronizacji, replikacji między hostami używając **zfs send** / **zfs recv.**

Pierwszy jail z SSH z użyciem iocage

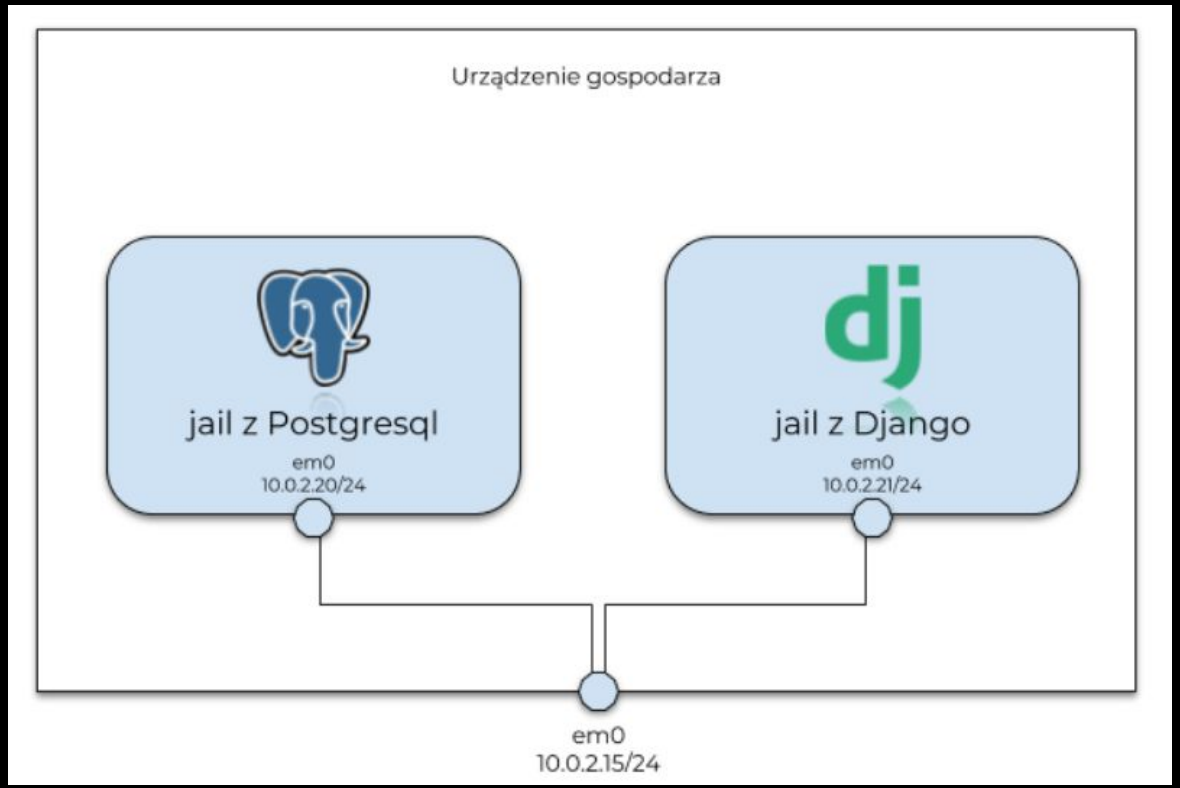
```
pkg install py36-iocage                                  # instalacja iocage z paczek

iocage fetch                                             # pobranie źródeł FreeBSD
iocage activate <poolname>                               # wskaznie poola dla nowych jaili
iocage create -r 11.1-RELEASE                            # stworzenia jaila dla zfetchowanego BSD
iocage list                                              # lista wszystkich jaili
iocage set ip4_addr="em0|10.1.1.10/24" <name># ustawienie adresu IP

iocage restart <name>                                    # restart jaila
iocage console <name>                                    # restart jaila
```

CZĘŚĆ PRAKTYCZNA

Wprowadzenie do części laboratoryjnej



- jail z bazą danych Postgresql
- jail z Django
- aplikacja webowa będzie wykorzystywać bazę do działania

Obraz 1. Scenariusz laboratorium.
Źródło: Opracowanie własne